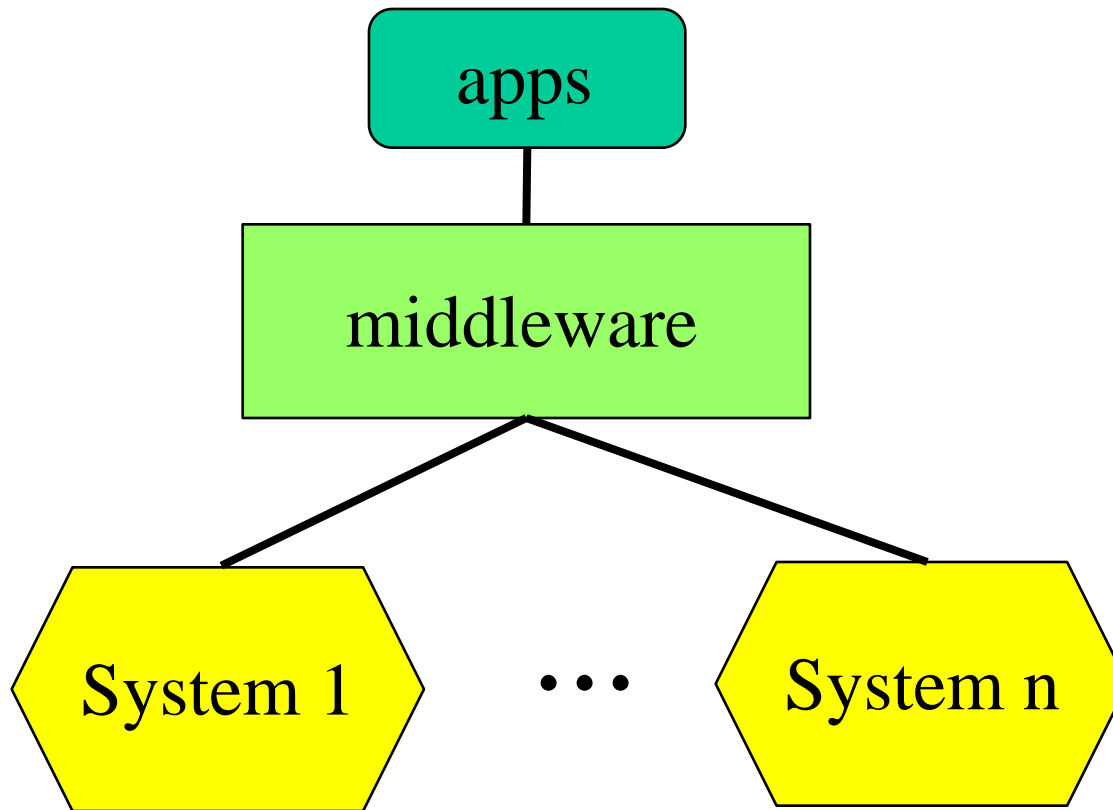


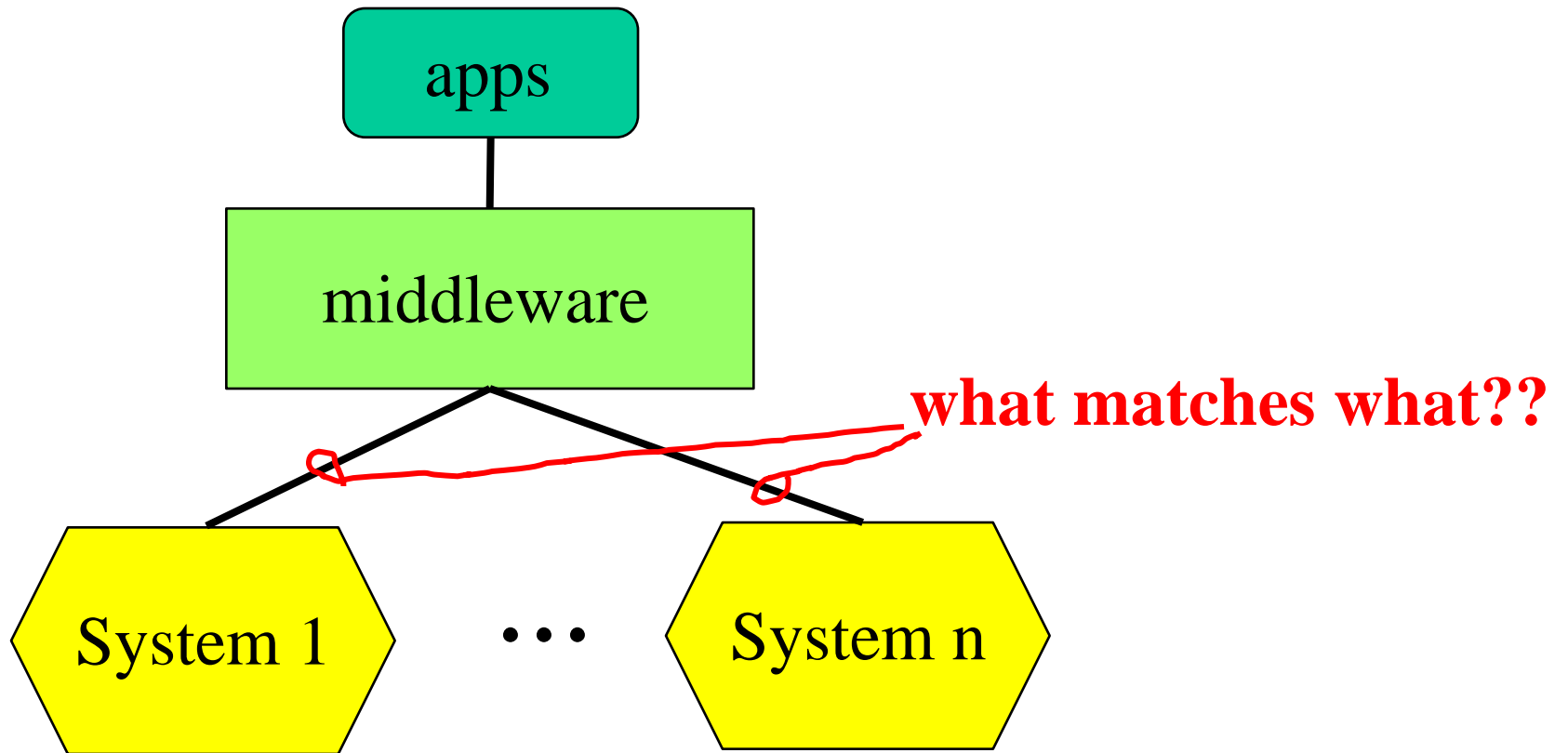
Entity Resolution: Glue for Middleware

*Hector Garcia-Molina
Stanford University*

Middleware



Middleware



Matching

- Execution Level
 - matching ports, calls, parameters, workflows ...
- Data Level
 - matching records, attributes, values ...

Matching

- Execution Level
- Data Level
 - Ontology
 - Schema
 - Instance

Example: Stock Options

- Ontology

Strike Price

Black–Scholes Valuation

Stock Option

Market Valuation

Stock Grant

Deferred Compensation

Date of Option

Taxable Income

Example: Stock Options

- Schema

Stock_Option(Date, Price, Shares, Holder, Plan, ...)


Option(Date, StrikePrice, Shares, Employee, Restrictions, ...)

Example: Stock Options


- Instance



Option 1

 Name: Tom S. Smith
— Adr: 123 Main St
— Date:
— Shares:

Option 2

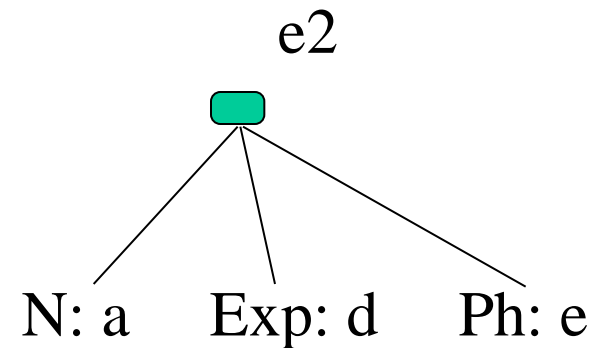
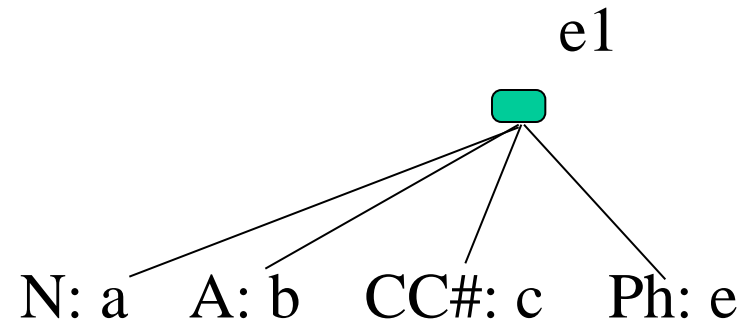
 Name: Thomas Smith
— Adr: 132 Main St
— Date:
— Shares:

This Talk

- Instance Resolution
 - a.k.a. Entity Resolution
 - a.k.a. De-Duplication
 - a.k.a. Record Linkage

Applications

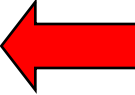
- comparison shopping
- mailing lists
- classified ads
- customer files
- counter-terrorism



Why is ER Challenging?

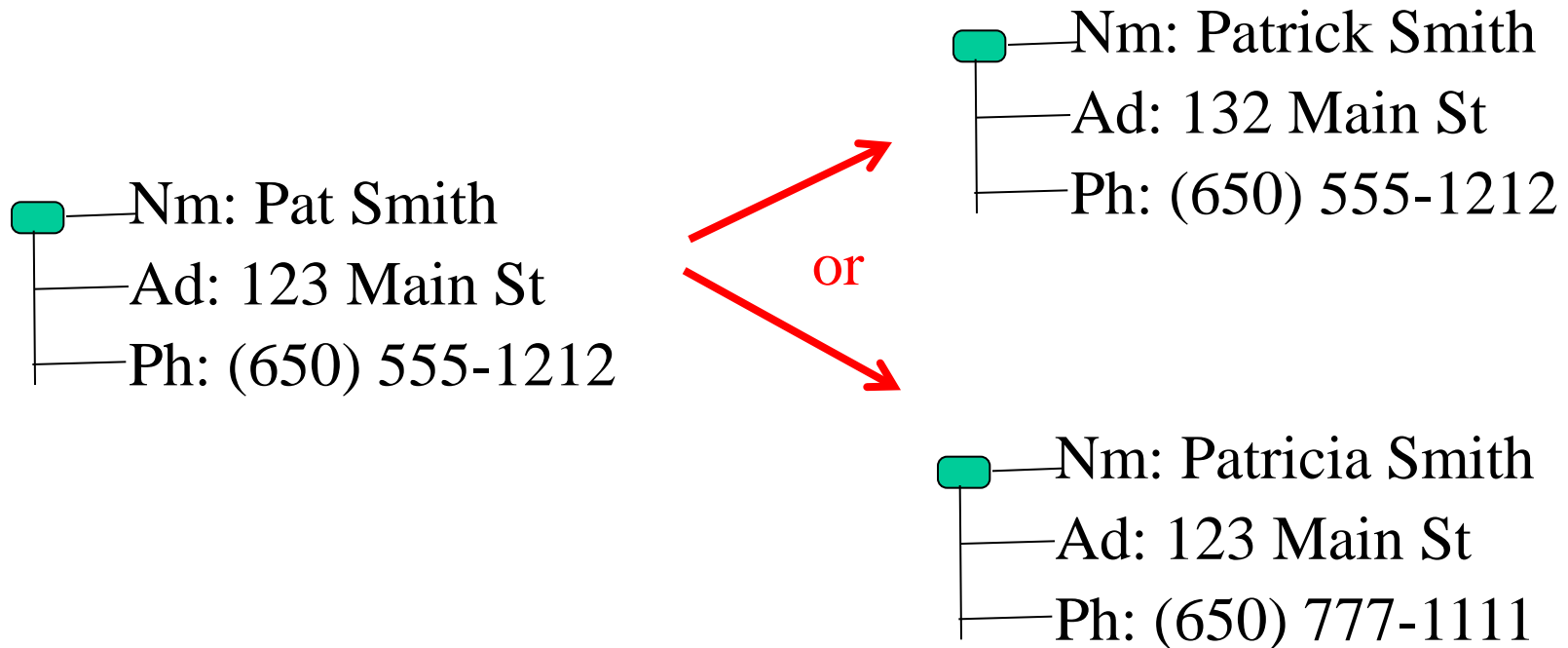
- Huge data sets
- No unique identifiers
- Lots of uncertainty
- Many ways to skin the cat

Outline

- Taxonomy 
- Swoosh Algorithm
- Distributed ER
- More on blocking

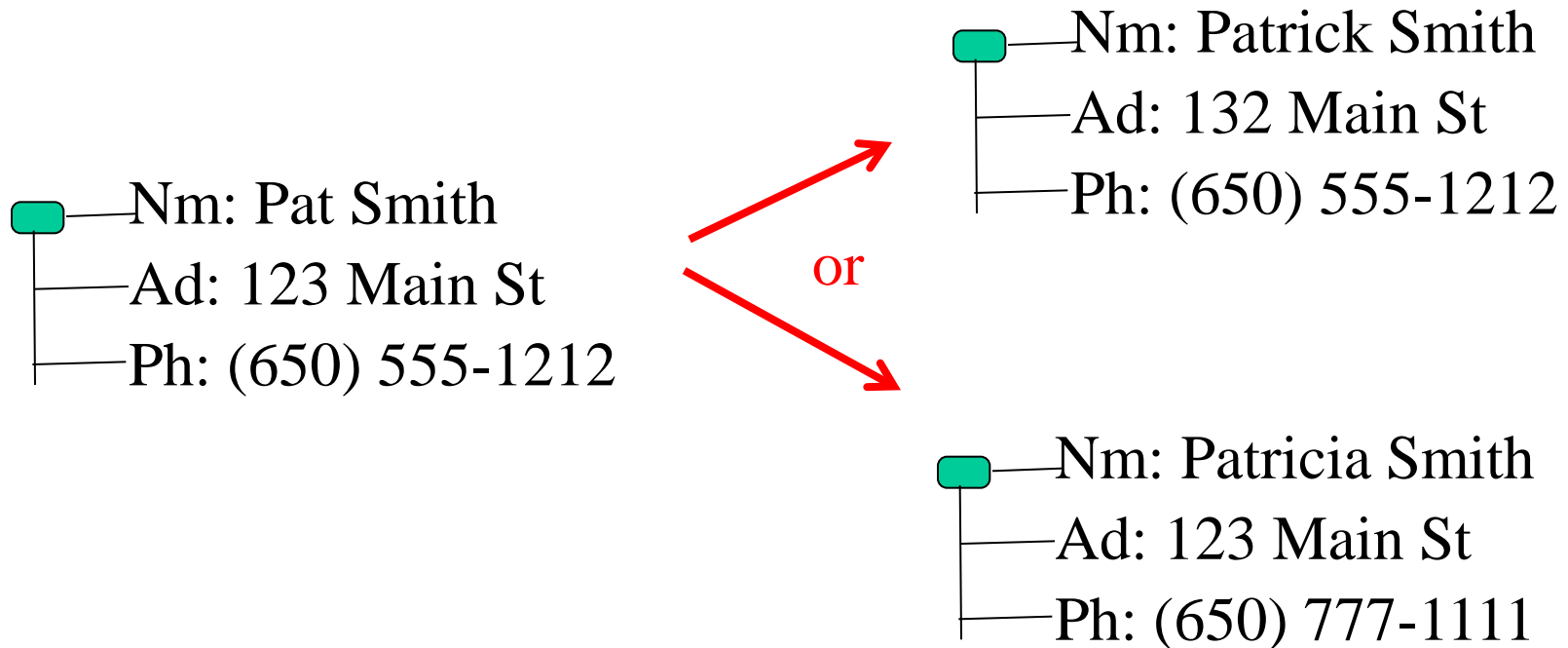
Taxonomy: Pairwise vs Global

- Decide if r , s match only by looking at r , s ?
- Or need to consider more (all) records?



Taxonomy: Pairwise vs Global

- Global matching complicates things a lot!
 - e.g., change decision as new records arrive

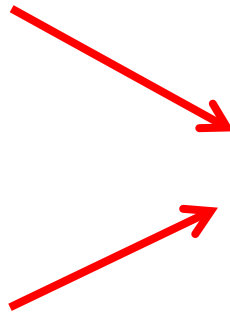


Taxonomy: Outcome

- Partition of records
 - e.g., comparison shopping
- Merged records

■ Nm: Pat Smith
— Ad: 123 Main St
— Ph: (650) 555-1212

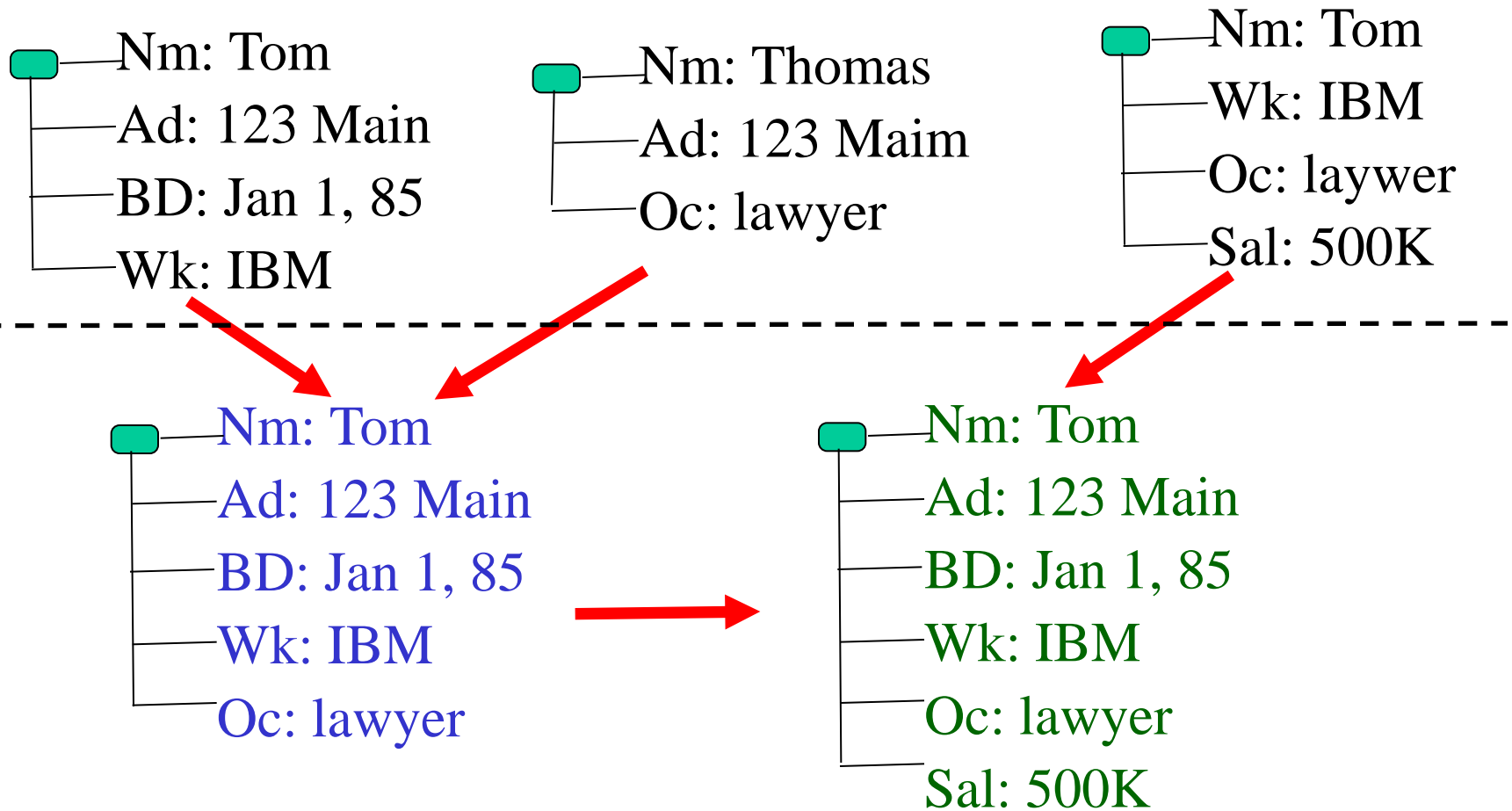
■ Nm: Patricia Smith
— Ad: 132 Main St
— Ph: (650) 777-1111
— Hair: Black



■ Nm: Patricia Smith
— Ad: 123 Main St
— Ph: (650) 555-1212
 (650) 777-1111
— Hair: Black

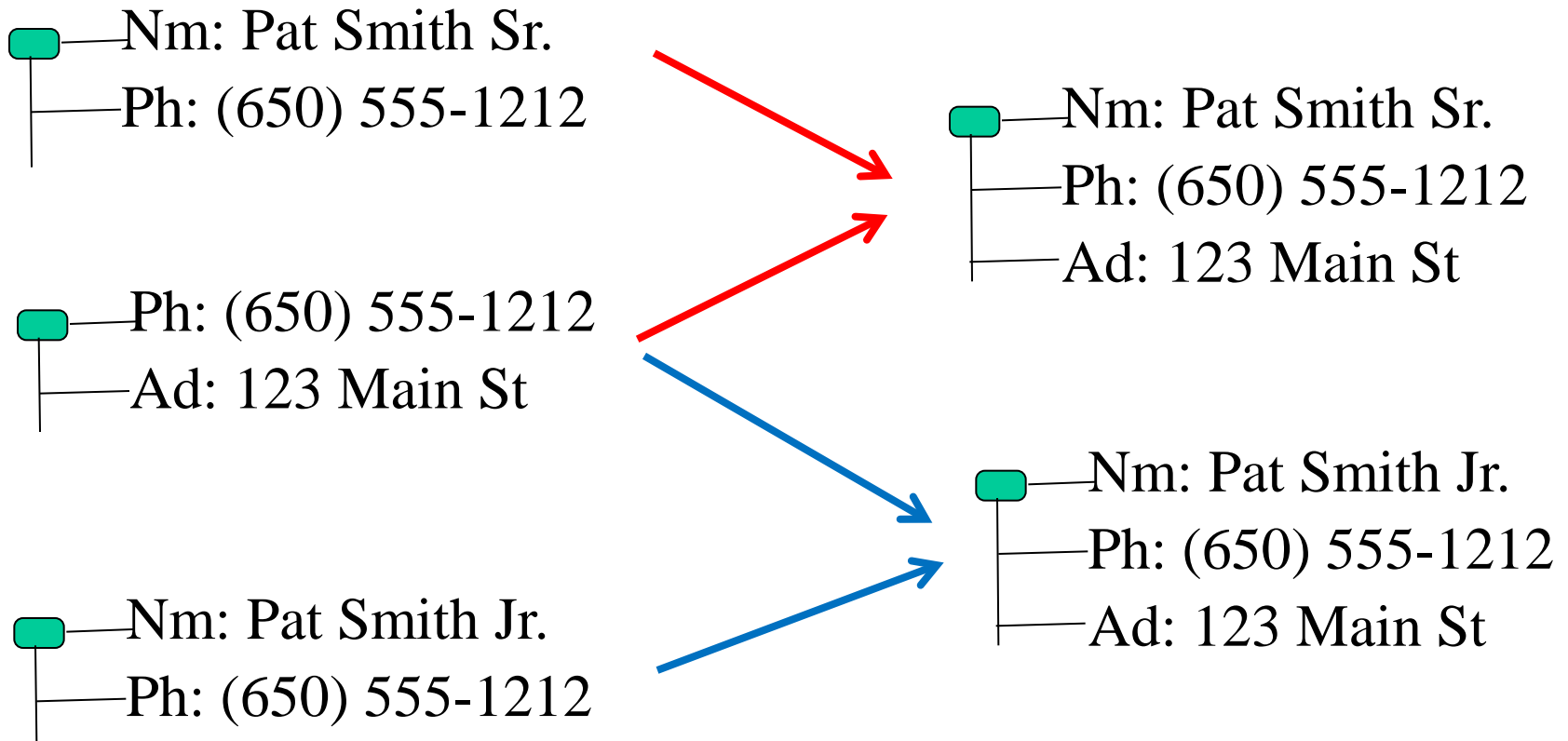
Taxonomy: Outcome

- Iterate after merging



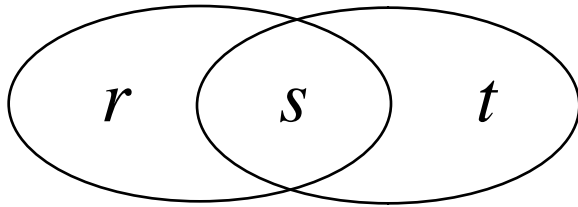
Taxonomy: Record Reuse

- One record related to multiple entities?

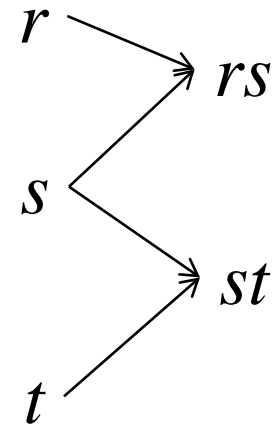


Taxonomy: Record Reuse

- Partitions

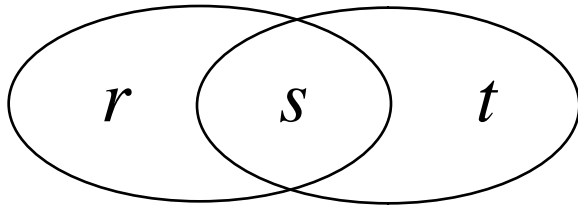


- Merges

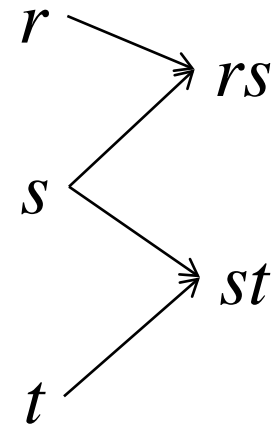


Taxonomy: Record Reuse

- Partitions

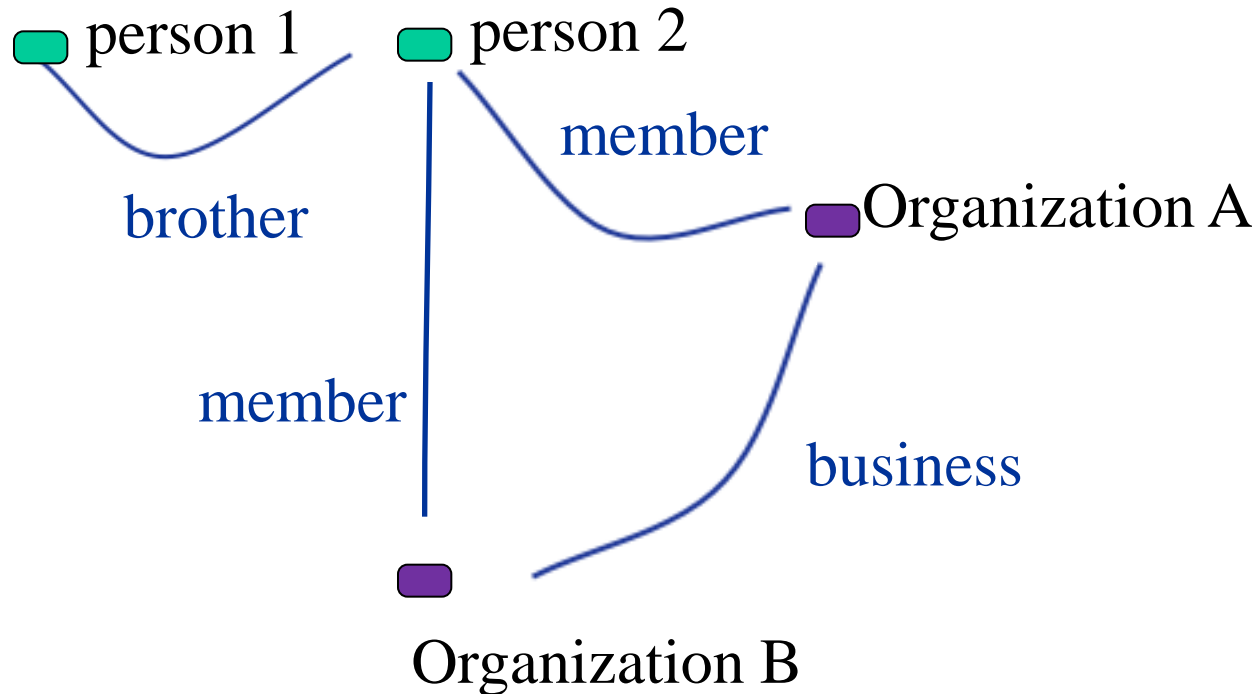


- Merges

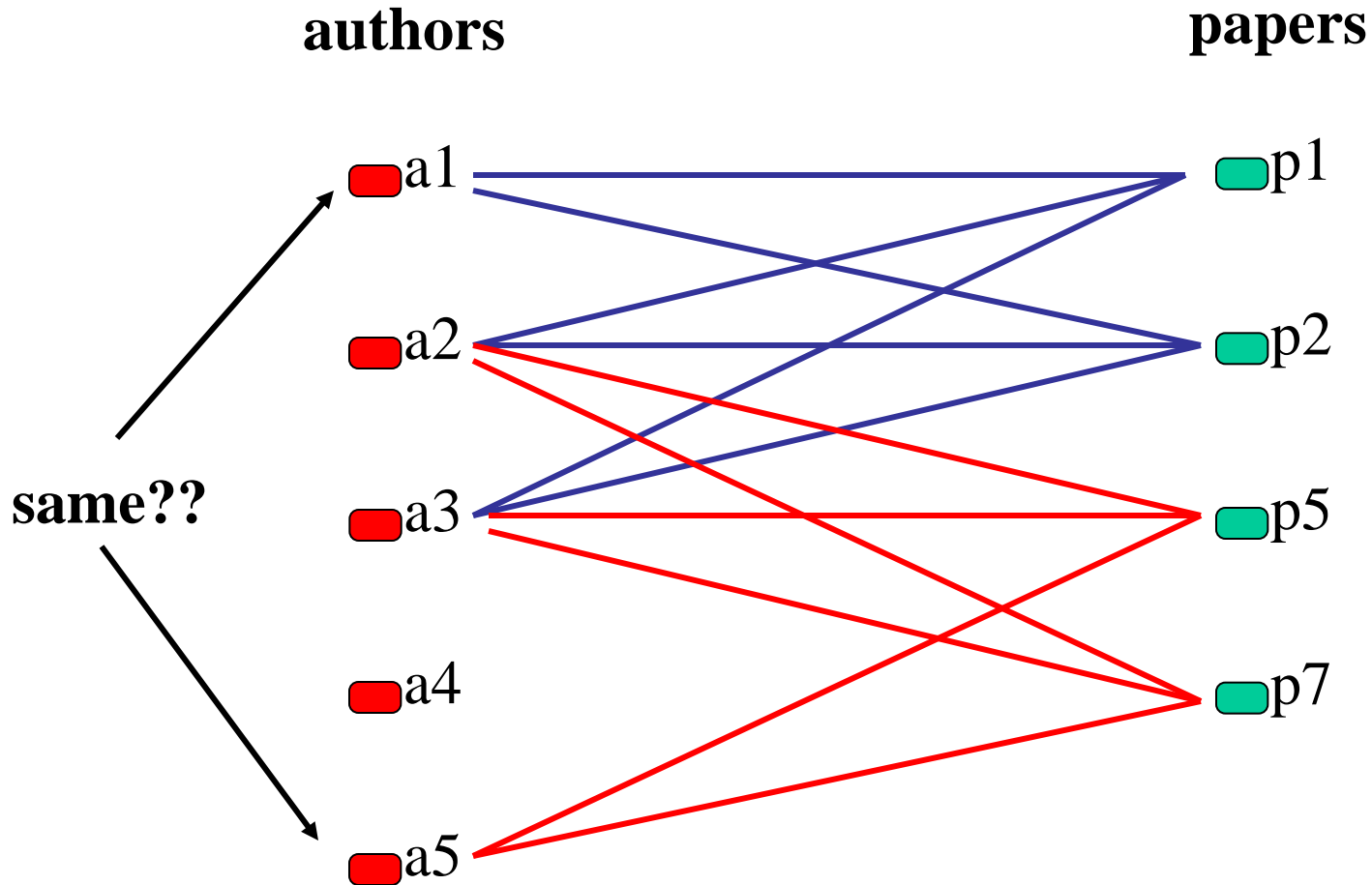


- Record reuse \Rightarrow complex and expensive!

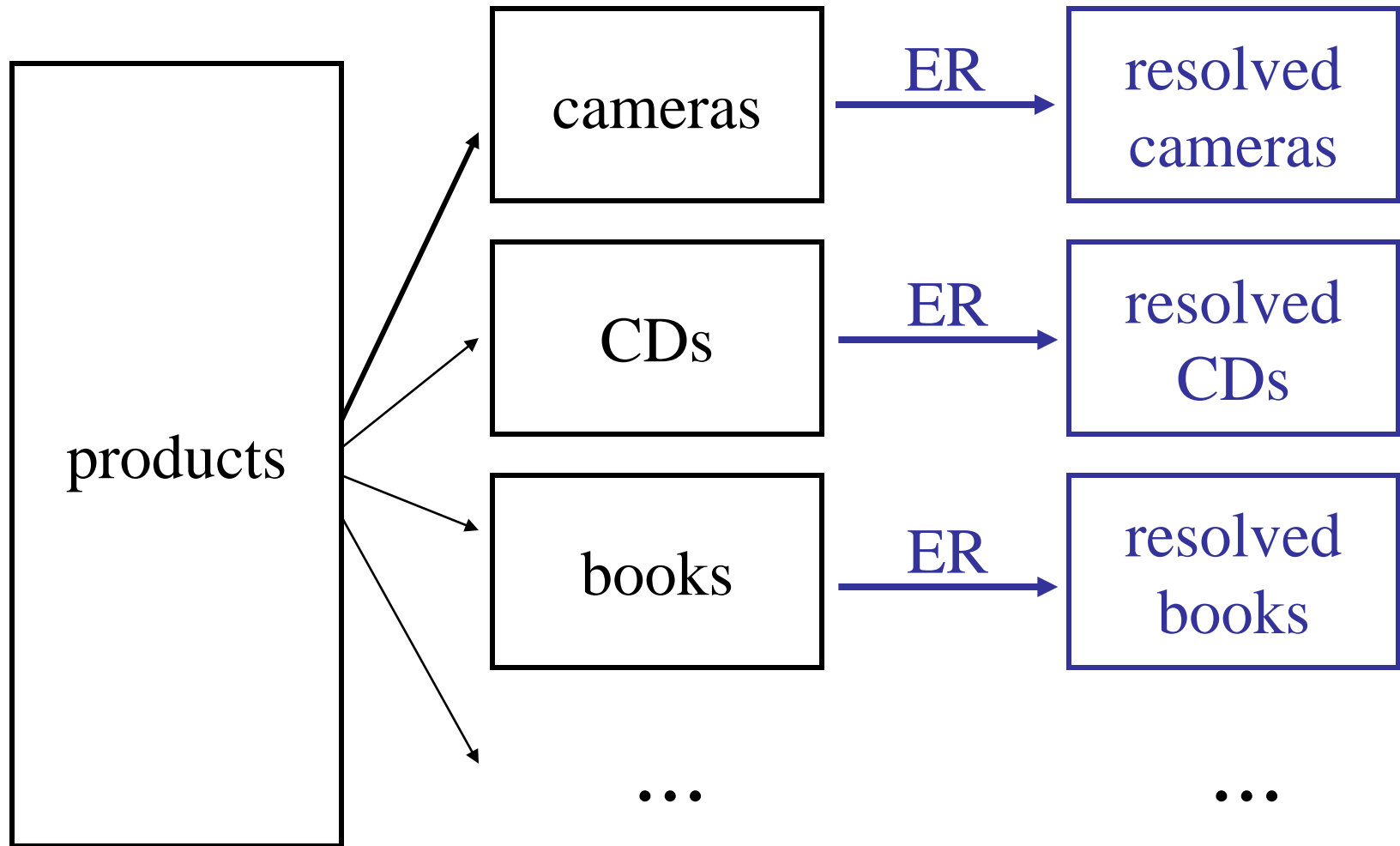
Taxonomy: Multiple Entity Types



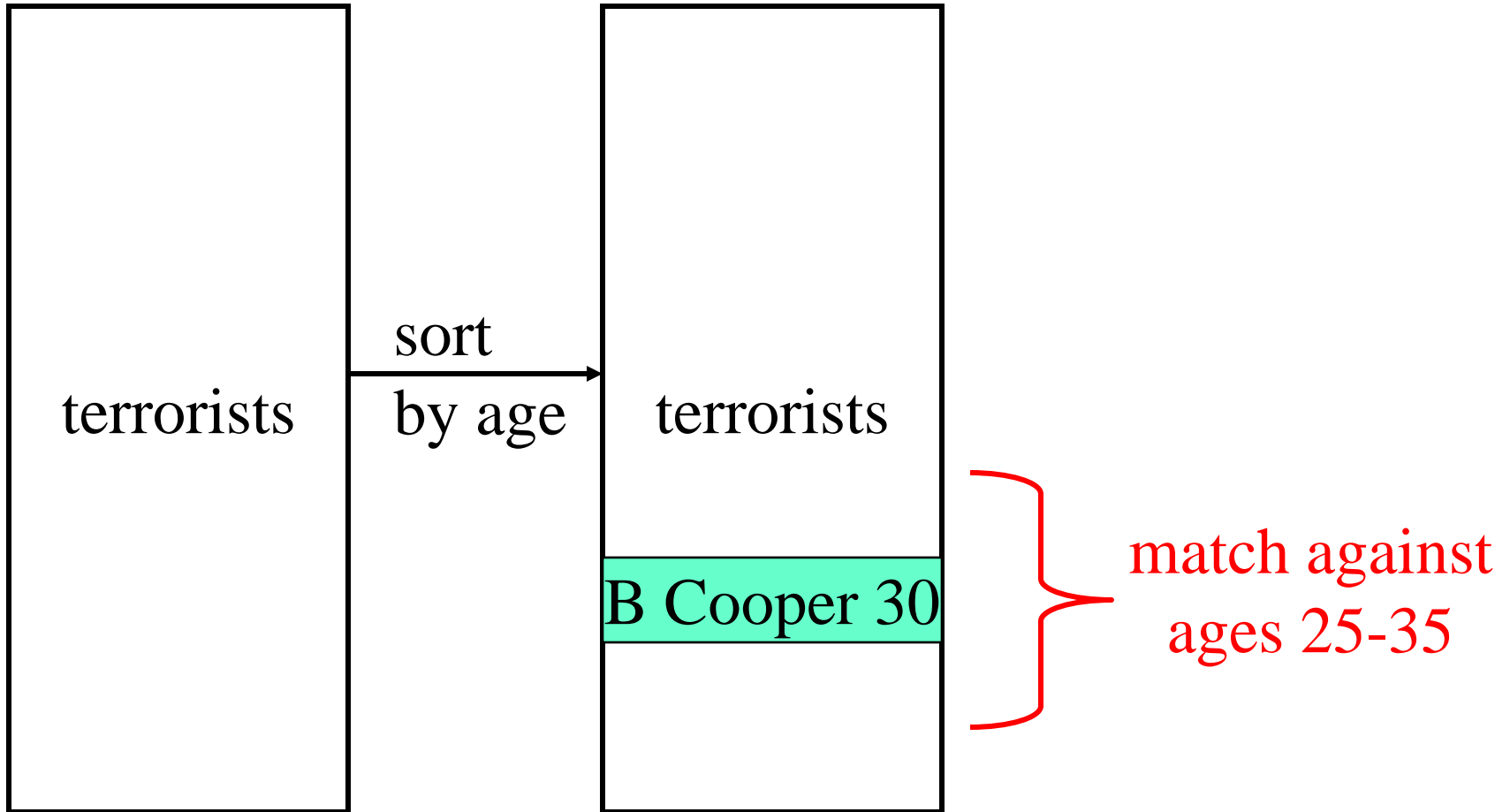
Taxonomy: Multiple Entity Types



Taxonomy: Exact vs Approximate



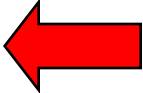
Taxonomy: Exact vs Approximate



Taxonomy: Other Variations

- Managing uncertainty
- Similarity computation

Outline

- Taxonomy
- Swoosh Algorithm 
- Distributed ER
- More on blocking

Scenario

- Pairwise matching
- Record merging
- No record reuse
- Single entity type

Model

r1

Nm: Tom
Ad: 123 Main
BD: Jan 1, 85
Wk: IBM

r2

Nm: Thomas
Ad: 123 Maim
Oc: lawyer

r3

Nm: Tom
Wk: IBM
Oc: lawyer
Sal: 500K

M(r1, r2)

Nm: Tom
Ad: 123 Main
BD: Jan 1, 85
Wk: IBM
Oc: lawyer

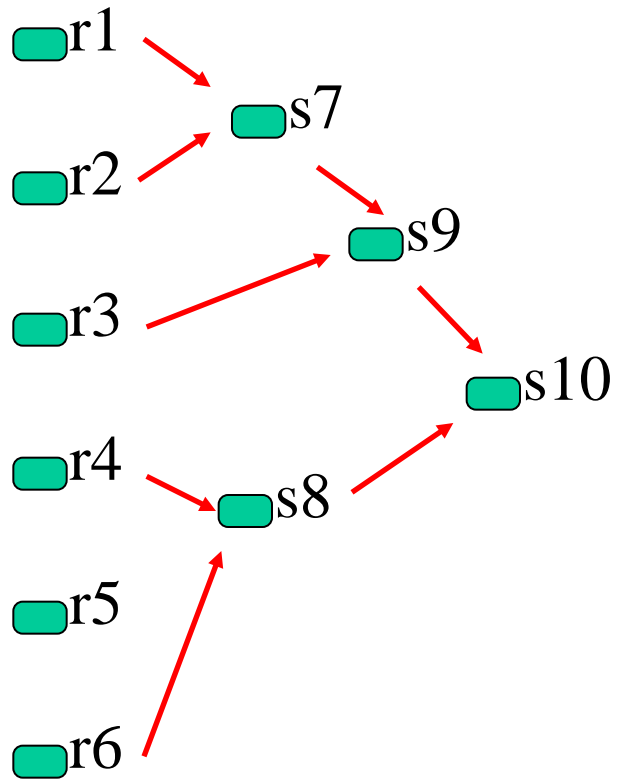
r4:<r1, r2>

M(r4, r3)

Nm: Tom
Ad: 123 Main
BD: Jan 1, 85
Wk: IBM
Oc: lawyer
Sal: 500K

<r4, r3>

Correct Answer



ER(R) = All derivable records.....

Minus “dominated” records

Question

- What is best sequence of match, merge calls that give us right answer?

Brute Force Algorithm

- Input R:
 - $r1 = [a:1, b:2]$
 - $r2 = [a:1, c:4, e:5]$
 - $r3 = [b:2, c:4, f:6]$
 - $r4 = [a:7, e:5, f:6]$

Brute Force Algorithm

- Input R:

- r1 = [a:1, b:2]

- r2 = [a:1, c: 4, e:5]

- r3 = [b:2, c:4, f:6]

- r4 = [a:7, e:5, f:6]

- Match all pairs:

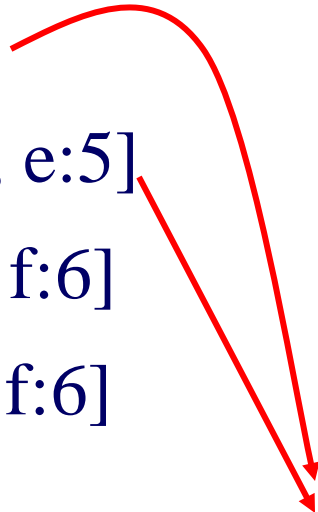
- r1 = [a:1, b:2]

- r2 = [a:1, c: 4, e:5]

- r3 = [b:2, c:4, f:6]

- r4 = [a:7, e:5, f:6]

- r12 = [a:1, b:2, c:4, e:5]



Brute Force Algorithm

- Match all pairs:

- $r1 = [a:1, b:2]$
- $r2 = [a:1, c:4, e:5]$
- $r3 = [b:2, c:4, f:6]$
- $r4 = [a:7, e:5, f:6]$
- $r12 = [a:1, b:2, c:4, e:5]$

- Repeat:

- $r1 = [a:1, b:2]$
- $r2 = [a:1, c:4, e:5]$
- $r3 = [b:2, c:4, f:6]$
- $r4 = [a:7, e:5, f:6]$
- $r12 = [a:1, b:2, c:4, e:5]$
- $r123 =$
 $[a:1, b:2, c:4, e:5, f:6]$

Question # 1

Brute Force Algorithm

- Input R:

- r1 = [a:1, b:2]

- r2 = [a:1, c: 4, e:5]

- r3 = [b:2, c:4, f:6]

- r4 = [a:7, e:5, f:6]

- Match all pairs:

- r1 = [a:1, b:2]

- r2 = [a:1, c: 4, e:5]

- r3 = [b:2, c:4, f:6]

- r4 = [a:7, e:5, f:6]

- r12 = [a:1, b:2, c:4, e:5]

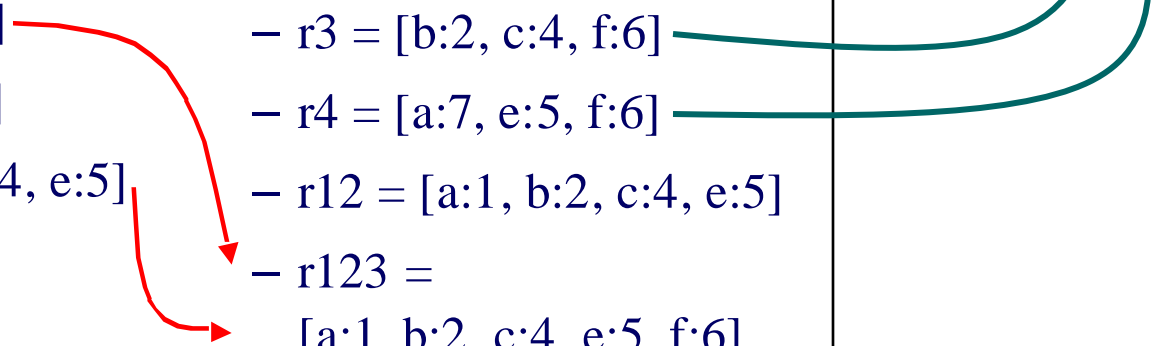
Can we delete
r1, r2?

Question # 2

Brute Force Algorithm

- Match all pairs:
 - r1 = [a:1, b:2]
 - r2 = [a:1, c: 4, e:5]
 - r3 = [b:2, c:4, f:6]
 - r4 = [a:7, e:5, f:6]
 - r12 = [a:1, b:2, c:4, e:5]
- Repeat:
 - r1 = [a:1, b:2]
 - r2 = [a:1, c: 4, e:5]
 - r3 = [b:2, c:4, f:6]
 - r4 = [a:7, e:5, f:6]
 - r12 = [a:1, b:2, c:4, e:5]
 - r123 = [a:1, b:2, c:4, e:5, f:6]

Can we avoid comparisons?

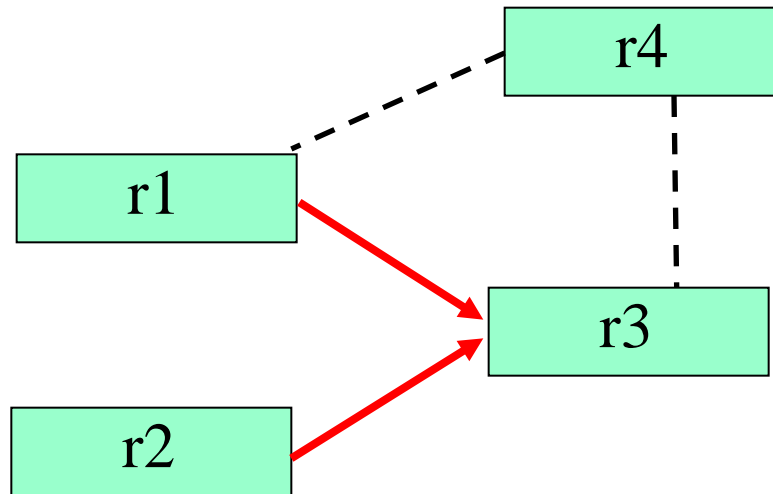


ICAR Properties

- Idempotence:
 - $M(r1, r1) = \text{true}; \langle r1, r1 \rangle = r1$
- Commutativity:
 - $M(r1, r2) = M(r2, r1)$
 - $\langle r1, r2 \rangle = \langle r2, r1 \rangle$
- Associativity
 - $\langle r1, \langle r2, r3 \rangle \rangle = \langle \langle r1, r2 \rangle, r3 \rangle$

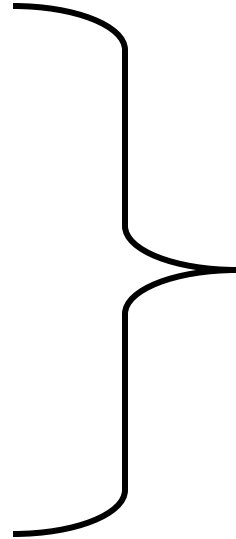
More Properties

- Representativity
 - If $\langle r1, r2 \rangle = r3$, then
for any $r4$ such that $M(r1, r4)$ is true
we also have $M(r3, r4) = \text{true}$.



ICAR Properties → Efficiency

- Commutativity
- Idempotence
- Associativity
- Representativity

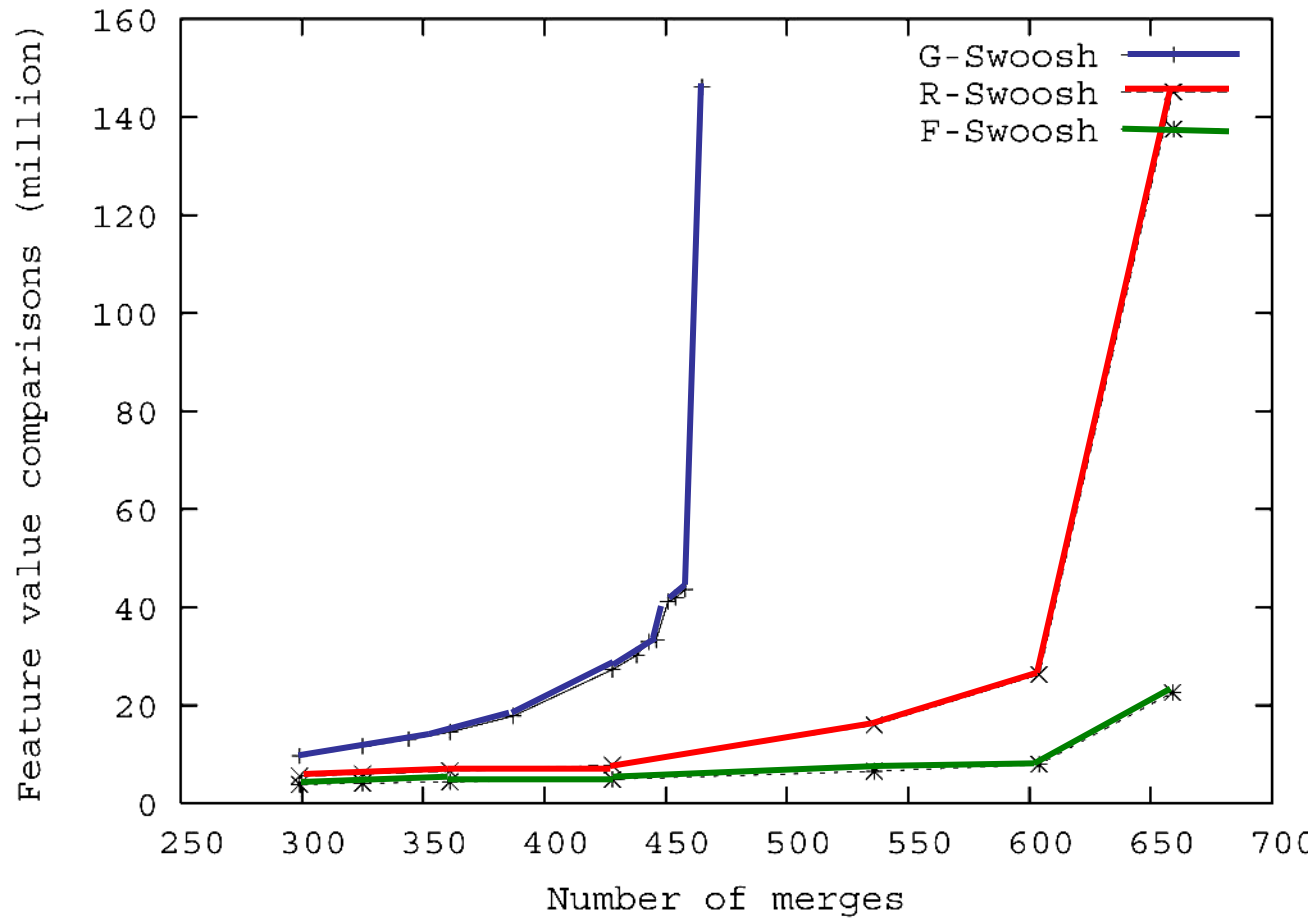


- Can discard records
- ER result independent of processing order

Swoosh Algorithms

- Record Swoosh
 - Merges records as soon as they match
 - Optimal in terms of record comparisons
- Feature Swoosh
 - Remembers values seen for each feature
 - Avoids redundant value comparisons

Swoosh Performance



If ICAR Properties Do Not Hold?

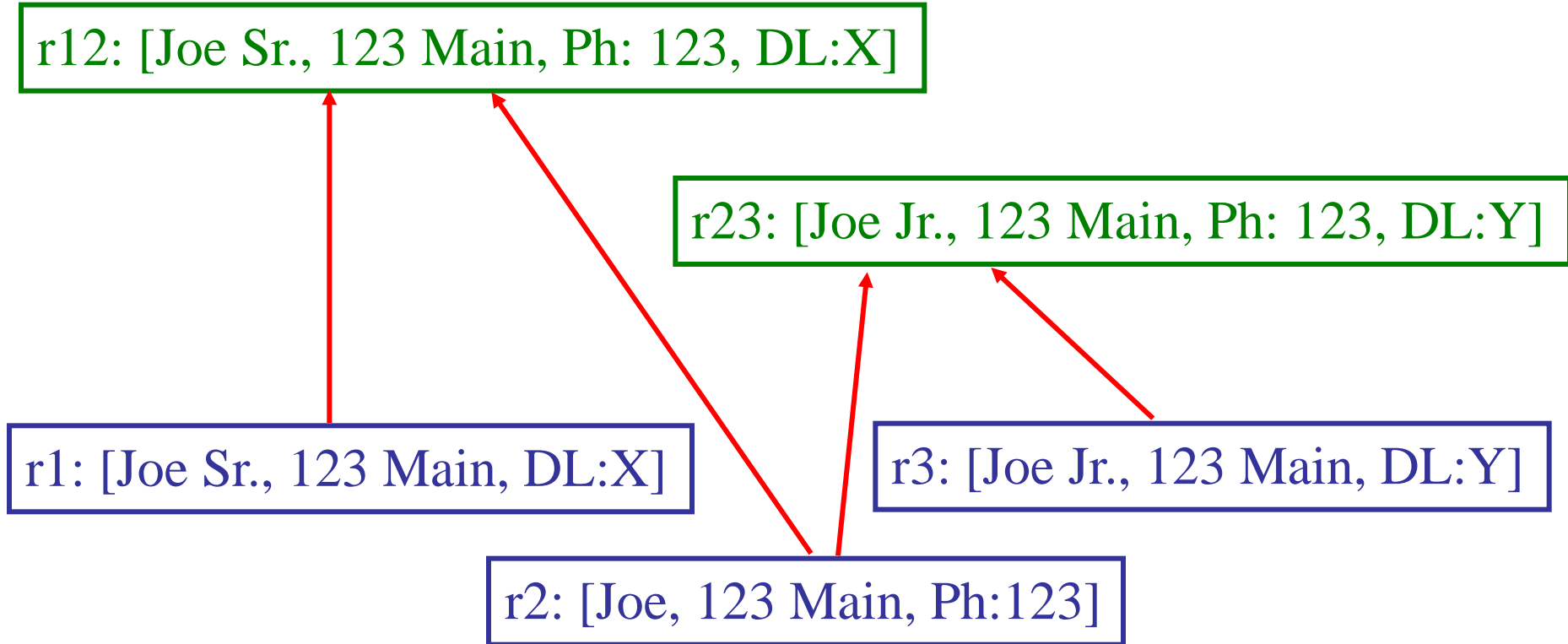
r12: [Joe Sr., 123 Main, Ph: 123, DL:X]

r23: [Joe Jr., 123 Main, Ph: 123, DL:Y]

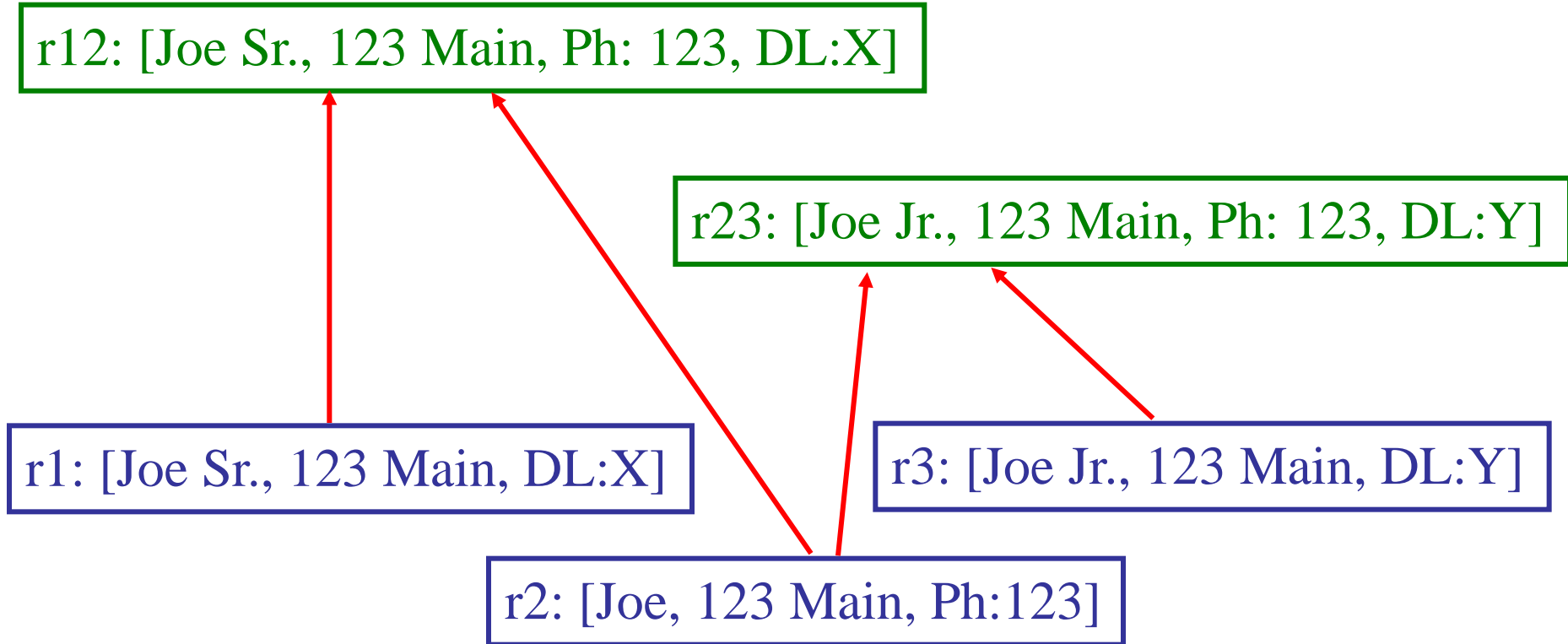
r1: [Joe Sr., 123 Main, DL:X]

r3: [Joe Jr., 123 Main, DL:Y]

r2: [Joe, 123 Main, Ph:123]



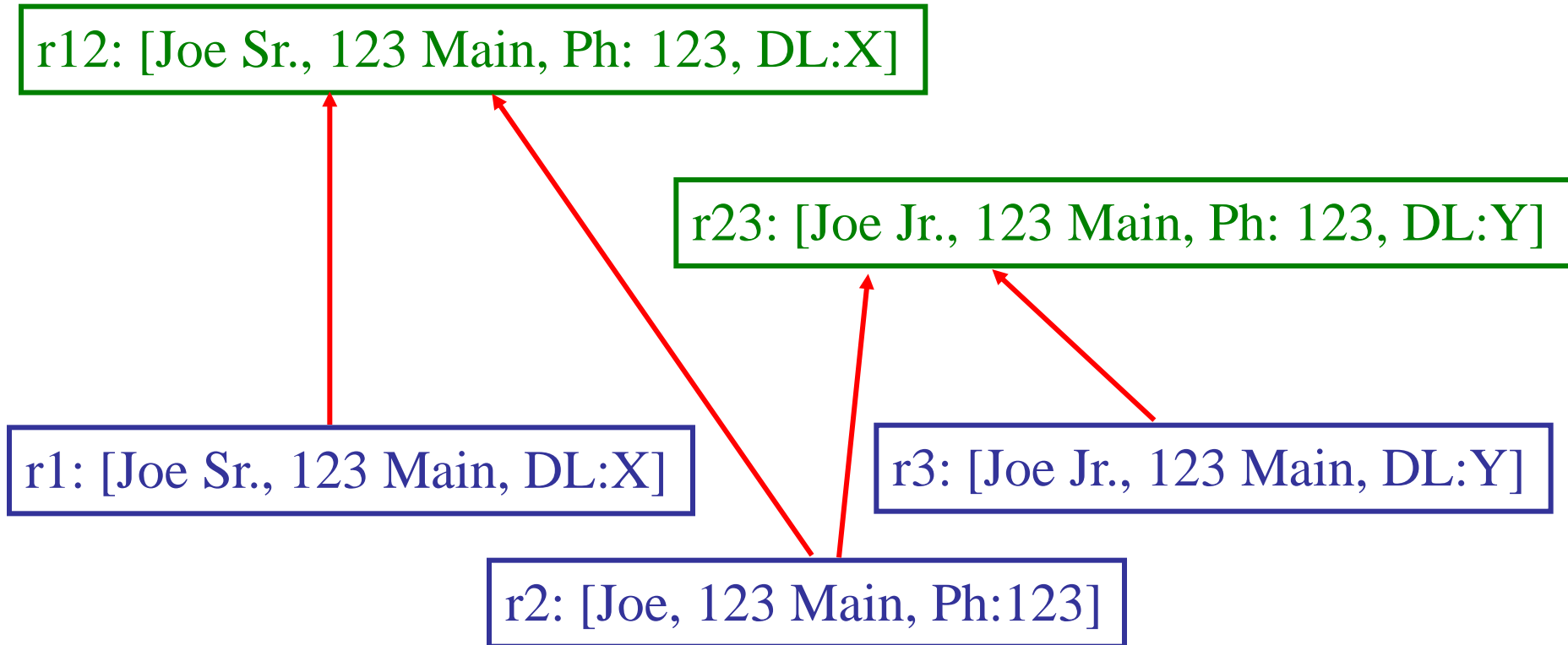
If ICAR Properties Do Not Hold?



Full Answer: $ER(R) = \{r12, r23, r1, r2, r3\}$

Minus Dominated: $ER(R) = \{r12, r23\}$

If ICAR Properties Do Not Hold?

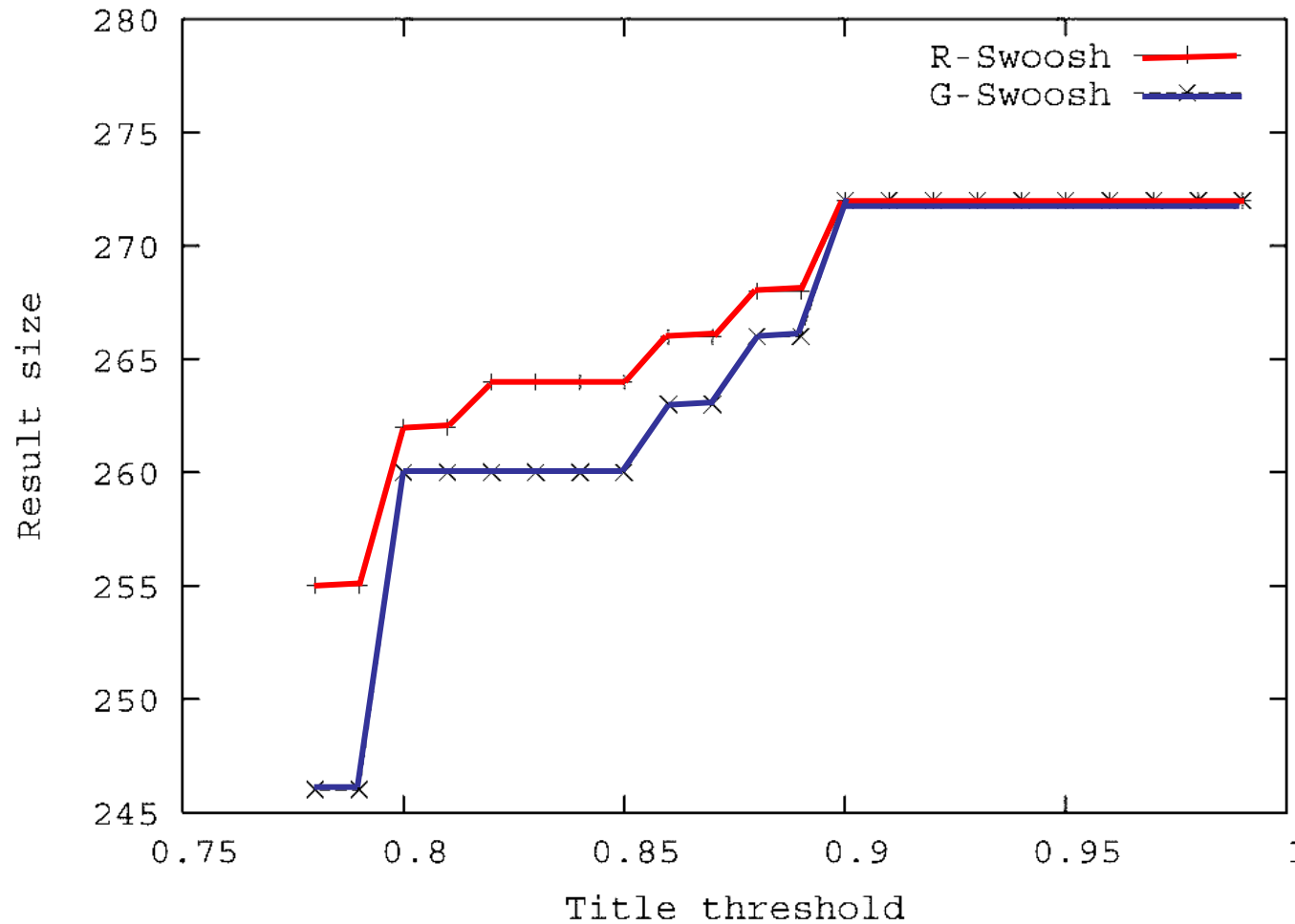


Full Answer: $ER(R) = \{r12, r23, r1, r2, r3\}$

Minus Dominated: $ER(R) = \{r12, r23\}$

R-Swoosh Yields: $ER(R) = \{r12, r3\}$ or $\{r1, r23\}$

Swoosh Without ICAR Properties



Distributed Swoosh

P1

P2

P3

r1
r2
r3
r4
r5
r6
...

Distributed Swoosh

P1

r1

r3

r4

r6

...

P2

r1

r2

r4

r5

...

P3

r2

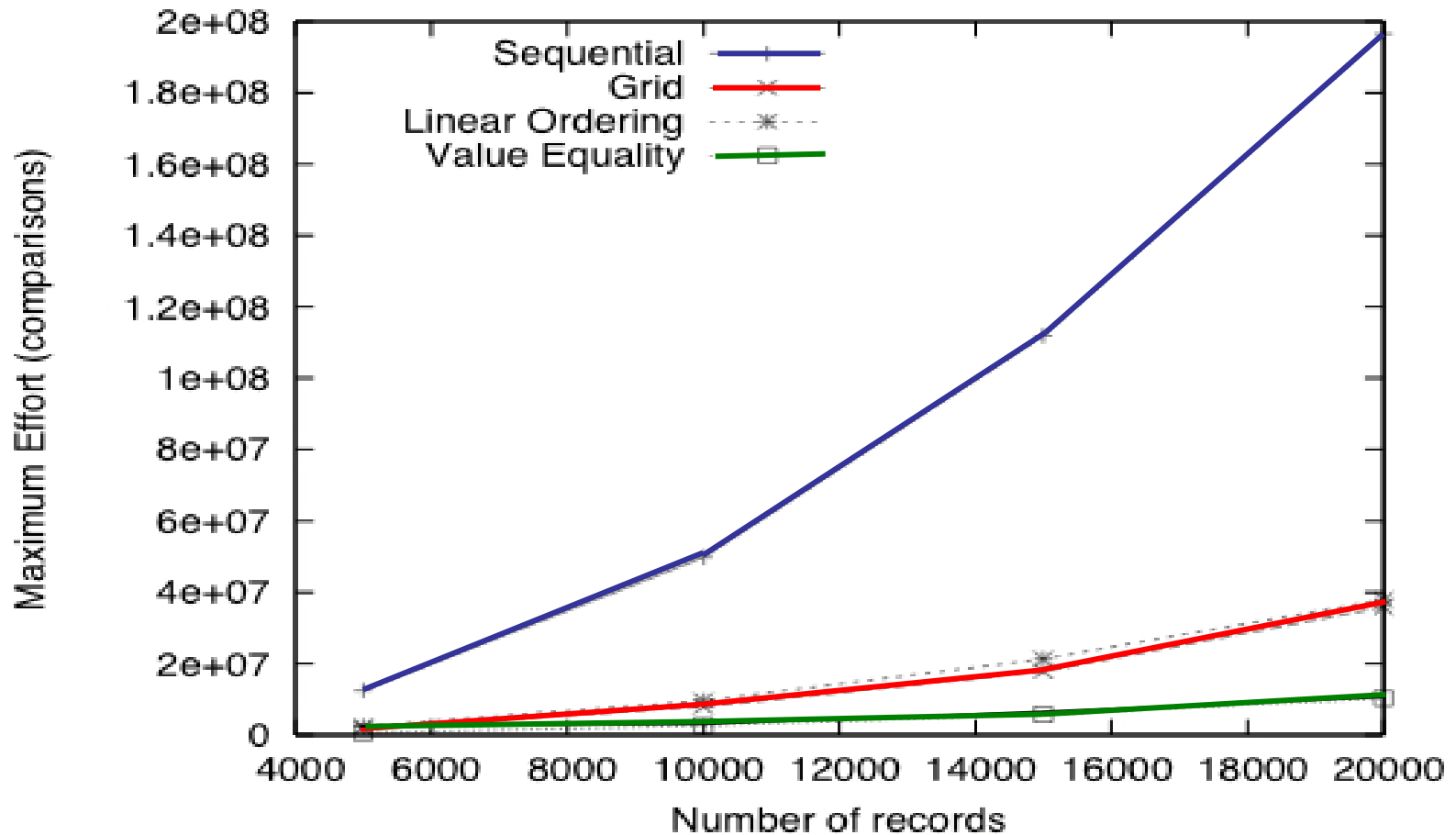
r3

r5

r6

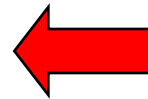
...

DSwoosh Performance



Outline

- Swoosh Algorithm
- Distributed ER
- More on blocking



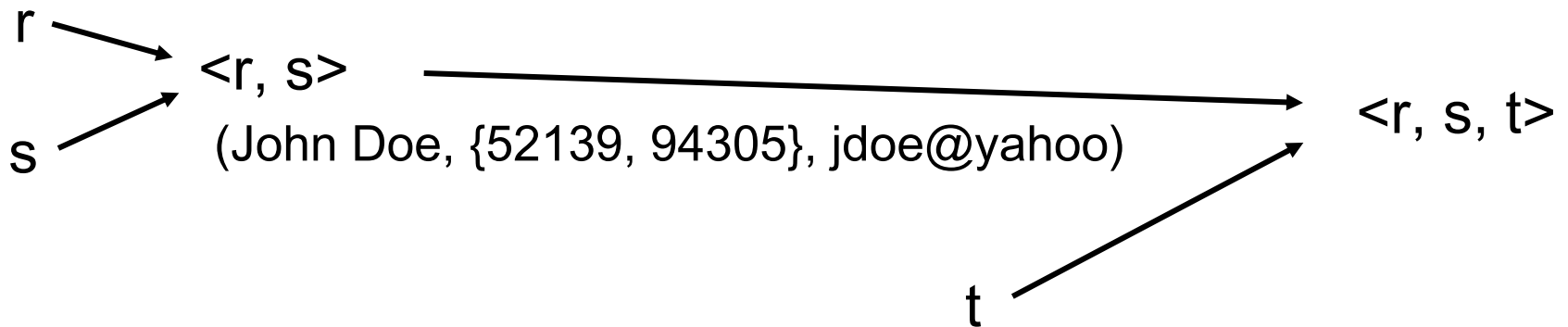
Iterative Blocking: Example

Record	Name	Addr (zip)	Email
r	John Doe	52139	jdoe@yahoo
s	John Doe	94305	
t	J. Foe	94305	jdoe@yahoo
u	Bobbie Brown	12345	bob@gmail
v	Bobbie Brown	12345	bob@gmail

Example

Record	Name	Addr (zip)	Email
r	John Doe	52139	jdoe@yahoo
s	John Doe	94305	
t	J. Foe	94305	jdoe@yahoo
u	Bobbie Brown	12345	bob@gmail
v	Bobbie Brown	12345	bob@gmail

Iterative ER:



Blocking

Record	Name	Addr (zip)	Email
r	John Doe	52139	jdoe@yahoo
s	John Doe	94305	
t	J. Foe	94305	jdoe@yahoo
u	Bobbie Brown	12345	bob@gmail
v	Bobbie Brown	12345	bob@gmail

Criterion	Partition by	$b_{-,1}$	$b_{-,2}$	$b_{-,3}$
SC1	zip code	r	s, t	u, v
SC2	1st char last name	r, s	t	u, v

Blocking

Record	Name	Addr (zip)	Email
r	John Doe	52139	jdoe@yahoo
s	John Doe	94305	
t	J. Foe	94305	jdoe@yahoo
u	Bobbie Brown	12345	bob@gmail
v	Bobbie Brown	12345	bob@gmail

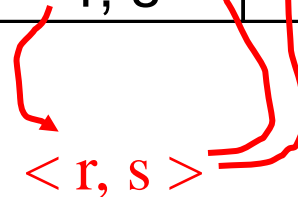
Criterion	Partition by	$b_{-,1}$	$b_{-,2}$	$b_{-,3}$
SC1	zip code	r	s, t	u, v
SC2	1st char last name	r, s	t	u, v

Will miss: $\langle r, s, t \rangle$

Blocking

Record	Name	Addr (zip)	Email
r	John Doe	52139	jdoe@yahoo
s	John Doe	94305	
t	J. Foe	94305	jdoe@yahoo
u	Bobbie Brown	12345	bob@gmail
v	Bobbie Brown	12345	bob@gmail

Criterion	Partition by	$b_{-,1}$	$b_{-,2}$	$b_{-,3}$
SC1	zip code	r	s, t	u, v
SC2	1st char last name	r, s	t	u, v

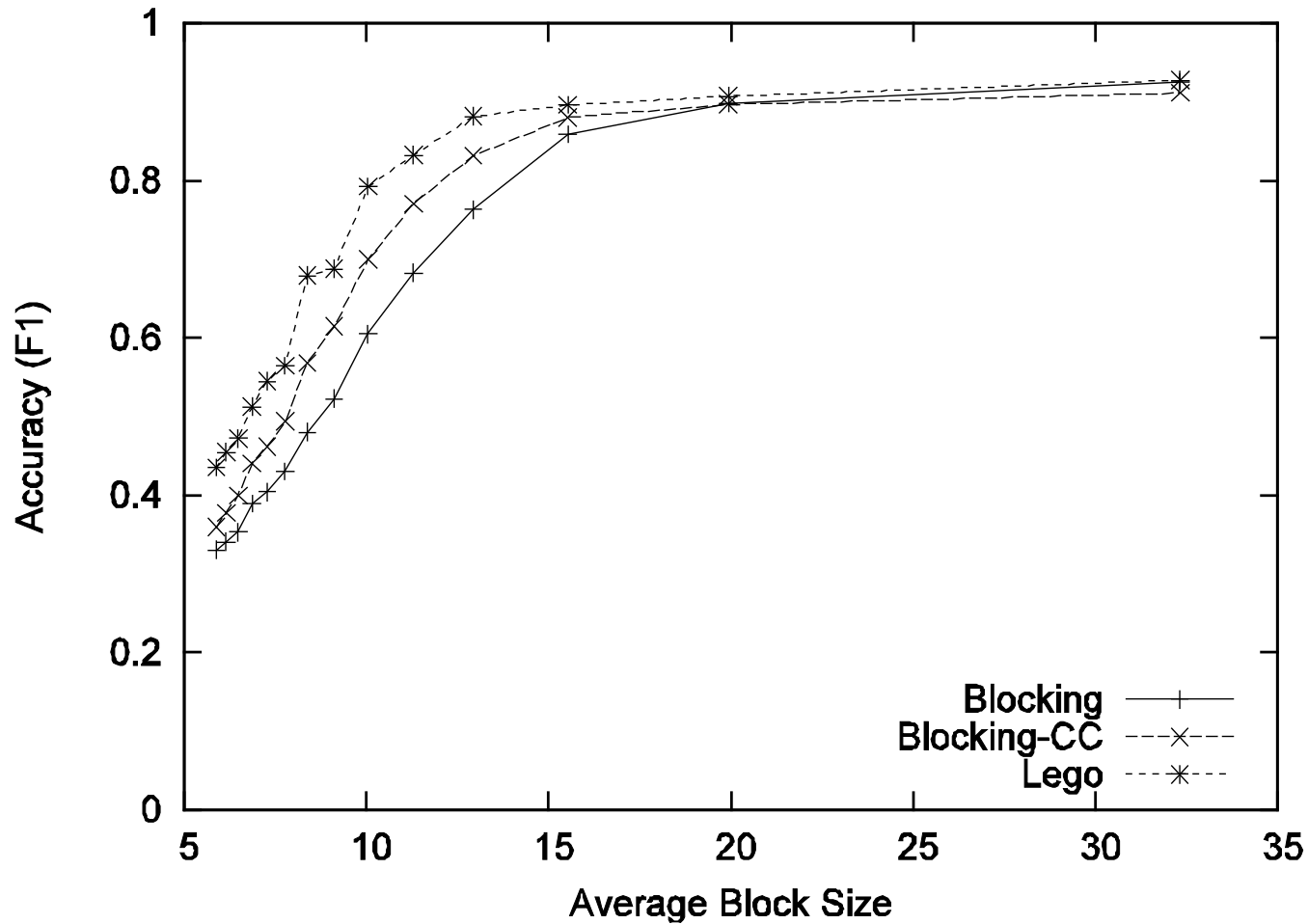


Solution: Propagate Matches

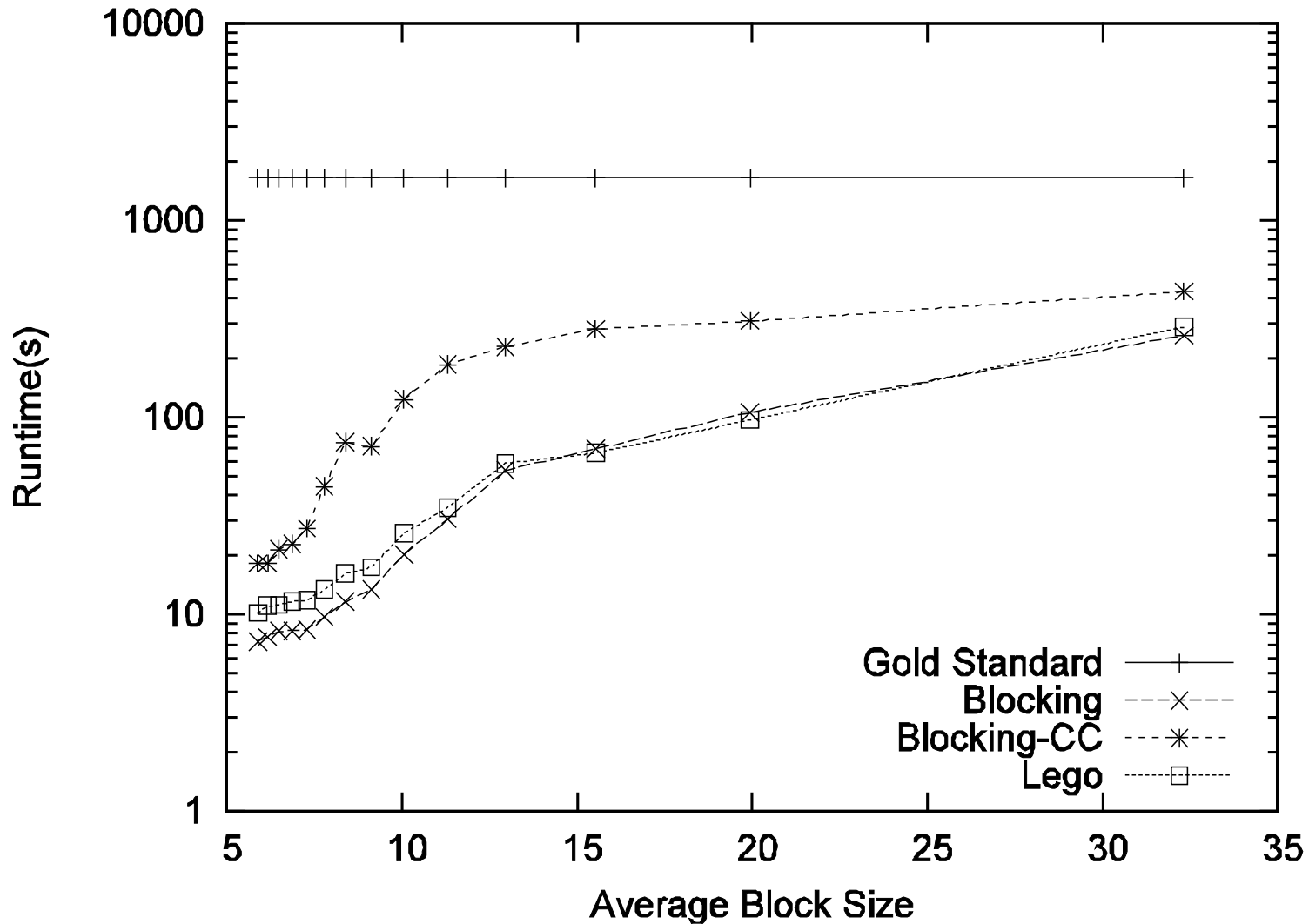
What We Have Done

- Formal Model for Iterative Blocking
 - based on generic “core” ER algorithm
- Two Algorithms:
 - Lego: in memory
 - Duplo: disk based
- Experiments

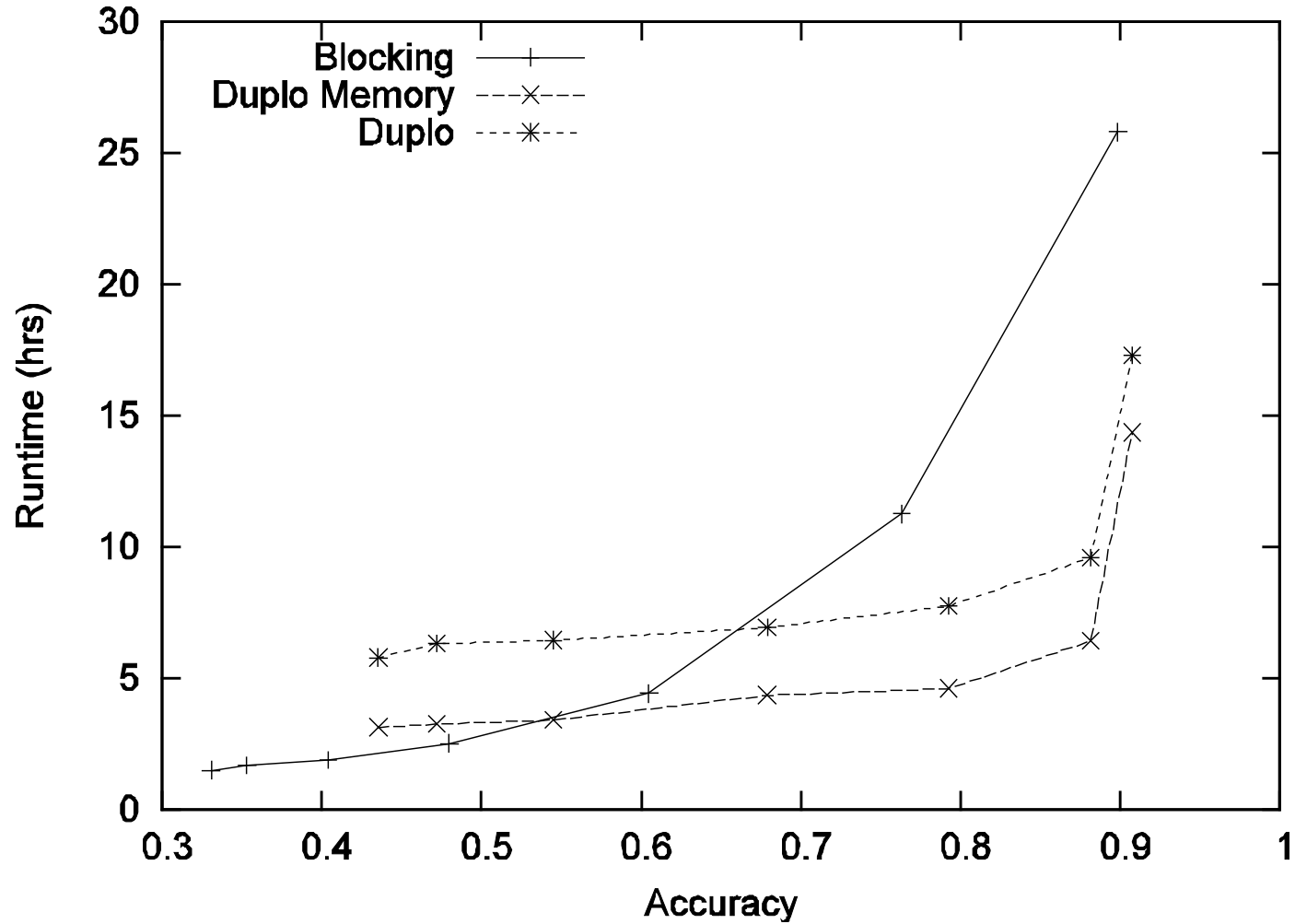
Sample Results: Accuracy



Sample Results: Run Time



Accuracy vs Run Time



Conclusion

- ER is old and important problem
- Critical for gluing components

Thanks.